G-4513 Series

3G Power Saving PAC with Solar Charger

User's Manual V1.0

High Quality, Industrial Data Acquisition, and Control Products

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assumes no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 2014 by ICP DAS CO., LTD. All rights reserved worldwide.

Trademark

The names used for identification only may be registered trademarks of their respective companies.

Contact us

If you have any problem, please feel free to contact us.
You can count on us for quick response.

Email：service@icpdas.com

Table of Contents

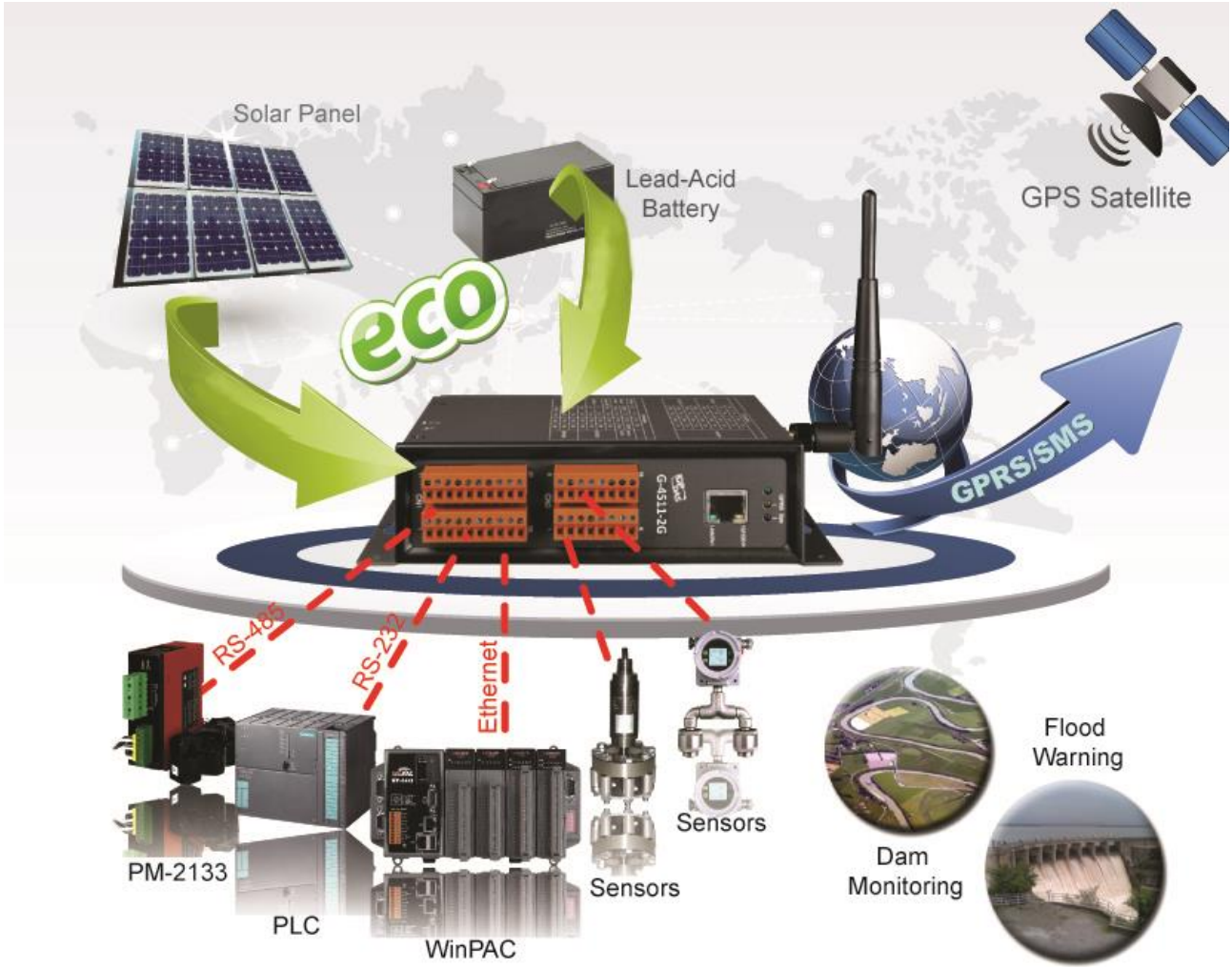
1. Introduction 1
2. Hardware Specifications 2
2.1 G-4513-3GWA Series 2
2.2 G-4513-3GWA Series Specifications 3
3. Application Architecture 6
3.1 Hydrologic / Wind Monitoring Application 6
3.2 Car Monitor / Tracking System 6
3.3 Redundancy Communication system 7
4. Hardware 8
4.1 Pin Assignments 8
4.2 Dimensions 9
4.3 Operation Mode Switch 10
4.4 LED Indicators 11
4.5 Wire Connection 12
4.6 Wake up from Sleep Mode 14
4.7 Installation 15
5. Power Saving and Charger 16
5.1 Power Saving 16
5.2 How to Choose the Battery 18
5.3 How to Choose the Solar Panel 21
6. Function Introduction 24
6.1 I/O Library Function Definition and Description 24
6.1.1 X305IO_Init 25
6.1.2 X305IO_GetLibVersion 26
6.1.3 X305IO_Read_AD_CalibrationGain 27
6.1.4 X305IO_Read_AD_CalibrationOffset 28
6.1.5 X305IO_AnalogIn 29
6.1.6 X305IO_Read_All_DI 30
6.1.7 X305IO_Read_One_DI 31
6.1.8 X305IO_Write_All_DO 32
6.1.9 X305IO_Write_One_DO 33
6.1.10 X305IO_Read_All_DO 34
6.1.11 X305IO_Read_One_DO 35
6.1.12 X305IO_AnalogIn_SetChannel 36
6.1.13 X305IO_AnalogIn_Hex 37
6.1.14 X305IO_AnalogIn_HexToFloat 38
6.2 MMC/SD Library Function Definition and Description 39
6.2.1 PC_Ertfs_Init 40

6.2.2 PC_Open 41
6.2.3 PC_Read 43
6.2.4 PC_Write 45
6.2.5 PC_Close 47
6.2.6 Get_ErrNo 48
6.2.7 PC_lseek 49
6.2.8 PC_MKDir 50
6.2.9 PC_RMDir 51
6.2.10 PC_deltree 52
6.2.11 PC_MV 53
6.2.12 PC_IsDir 54
6.2.13 PC_Pwd 55
6.2.14 PC_Get_Attributes 56
6.2.15 PC_Set_Attributes 58
6.3 LCD Library Function Definition and Description 60
6.3.1 LCD_Init 61
6.3.2 LCD_BackLight_On 62
6.3.3 LCD_BackLight_Off 63
6.3.4 LCD_ShowText 64
6.3.5 LCD_DisplayNumber 65
6.3.6 LCD_SetNumber 66
6.3.7 LCD_ClrNumber 67
6.3.8 LCD_ClrScrn 68
6.3.9 LCD_StandByMode 69
6.3.10 LCD_NormalMode 70
6.3.11 LCD_GotoPosition 71
6.3.12 LCD_CursorDisplay 72
6.3.13 LCD_LineReverse 73
6.3.14 LCD_LineRestore 74
6.3.15 LCD_GetLibDate 75
6.3.16 LCD_GetLibVersion 76
7. Program Download Procedure 77
8. Revision History 83

1. Introduction

The G-4513 series are M2M (Machine to Machine) Power Saving PAC with a cellular transceiver and Solar Charger. It can be used in hydrologic monitoring or mudslide monitoring system. With optional GPS model, the G-4513 can also be a GPS tracking system for vehicle management system or maritime system.

The features of G-4513 series: Solar Charger, 3G module, Ethernet interface, optional GPS module, 3 digital inputs, 3 digital outputs, 8 analog inputs, 1 relay, 1 RS-232 and 1 RS-485 port. That can be used in various application fields to transfer data over 3G, GPRS, SMS, Ethernet or serial bus. The G-4513 series built-in MiniOS7 provide the same development environment with I-7188/I-7186 series. It is easier for I-7188/I-7186 users to apply the G-4513 series.



2. Hardware Specifications

2.1 G-4513-3GWA Series

G-4513-3GWA	G-4513D-3GWA	G-4513P-3GWA	G-4513PD-3GWA

2.2 G-4513-3GWA Series Specifications

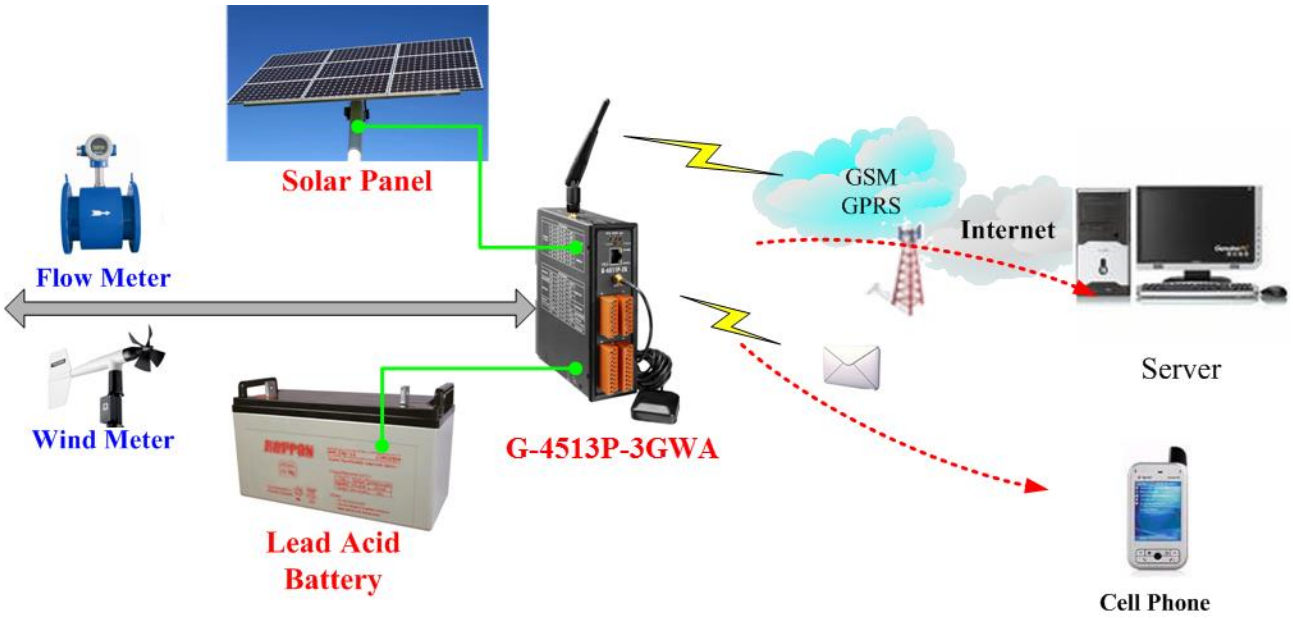
Item	G-4513-3GWA	G-4513D-3GWA	G-4513P-3GWA	G-4513PD-3GWA
CPU	80 MHz internal microprocessor			
SRAM/Flash	512K/512K , real time clock, watchdog timer			
NVRAM	31 bytes, battery backup, data valid up to 10 years			
EEPROM	16 KB, retention > 40 years. 1,000,000 erase/write cycles			
Comm. Interface				
COM ports	COM1:5-wire RS-232; COM2: RS-485			
Ethernet	10/100 Base-TX Ethernet controller			
GSM Interface				
Frequency Band	GSM 850/900/1800/1900 MHz			
GPRS connectivity	GPRS class 10/8; GPRS station class B			
SMS	MT, MO, CB, Text and PDU mode			
3G Interface				
Frequency Band	WCDMA 850/900/1900/2100 MHz			
Power Class	Class 3 (250mW @ WCDMA/HSPA)			
Digital Input				
Input Channel	3			
Input Type	Source(Dry Type), Common Ground			
Off Voltage Level	+1 V max.			
On Voltage Level	+3.5 ~ +30 V			
Isolated Voltage	Non-isolated			
Digital Output				
Output Channel	3			
Output Type	3 Open Collector (Sink/NPN)			
Load Voltage	+30 VDC max.			
Load Current	100 mA max.			
Isolated Voltage	Non-isolated			
Analog Input				
Input Channel	8			
Resolution	12 - bit			
Input Range/Type	0 ~ 20 mA			
Sample Rate	1 KHz max. (Read one channel)			
Accuracy	+/- 2 LSB (+/- 0.0097 mA)			
Isolated Voltage	2500Vrms 3000Dc to DC			
Relay				
Output Channel	1			
Type	Form C			

Input Range		2A@30 Vdc ; 0.25 A @250 Vac			
Mechanical endurance		typ. 10^8 operations			
GPS Interface					
Support Channels		-		32	
Sensitivity		-		Tracking = up to -159 dBm (with external LNA)	
				Cold start = up to -146 dBm (with external LNA)	
Acquisition Time		-		Hot start (Open Sky) = 2 s(typical)	
				Cold start (Open Sky) = 36 s(typical)	
Protocol Support		-		NMEA 0183 version 3.01	
LCD Interface					
General	Effective display area	-	80.61 mm x 14.37 mm (W x H)	-	80.61 mm x 14.37 mm (W x H)
	Module Dimension	-	93 mm x 70 mm x 1.6 mm (W x H x T)	-	93 mm x 70 mm x 1.6 mm (W x H x T)
Life Time		-	Expected life is more than 100,000 hours under normal operation	-	Expected life is more than 100,000 hours under normal operation
Power (Solar Input)					
Protection		Power reverse polarity protection			
Frame Ground Protection		ESD, Surge, EFT, Hi-Pot			
Power Requirement		+10 V_{DC} ~ +30 V_{DC} , (Max. Voltage of Solar Panel must less +30V)			
Power Consumption		Sleep: < 10 mA@12V_{DC}; Idle: 77 mA @ 24 V_{DC}; Data Link: 150 ~ 400 mA (peak) @ 24 V_{DC}			
Lead Acid Battery Requirement					
Battery		12V Lead-Acid Battery			
Charging Voltage		Voltage of Power Input must be over +16V			
Low Voltage Protect		Low Voltage disconnect = 11.1V / Low Voltage reconnect = 12.6V			
LED Indicators					
System		Red			
3G		Yellow			
GPS		-		Green	

Charging / Fault	Green / Red			
Mechanical				
Casing	Metal			
Dimensions	47 mm x 142 mm x 168 mm (W x L x H)			
Installation	DIN-Rail and Wall mount			
Environment				
Operating Temperature	-20 ~ +70 °C	-15 ~ +55 °C	-20 ~ +70 °C	-15 ~ +55 °C
Storage Temperature	-40 ~ +80 °C	-20 ~ +70 °C	-40 ~ +80 °C	-20 ~ +70 °C
Humidity	5~90% RH, non-condensing			

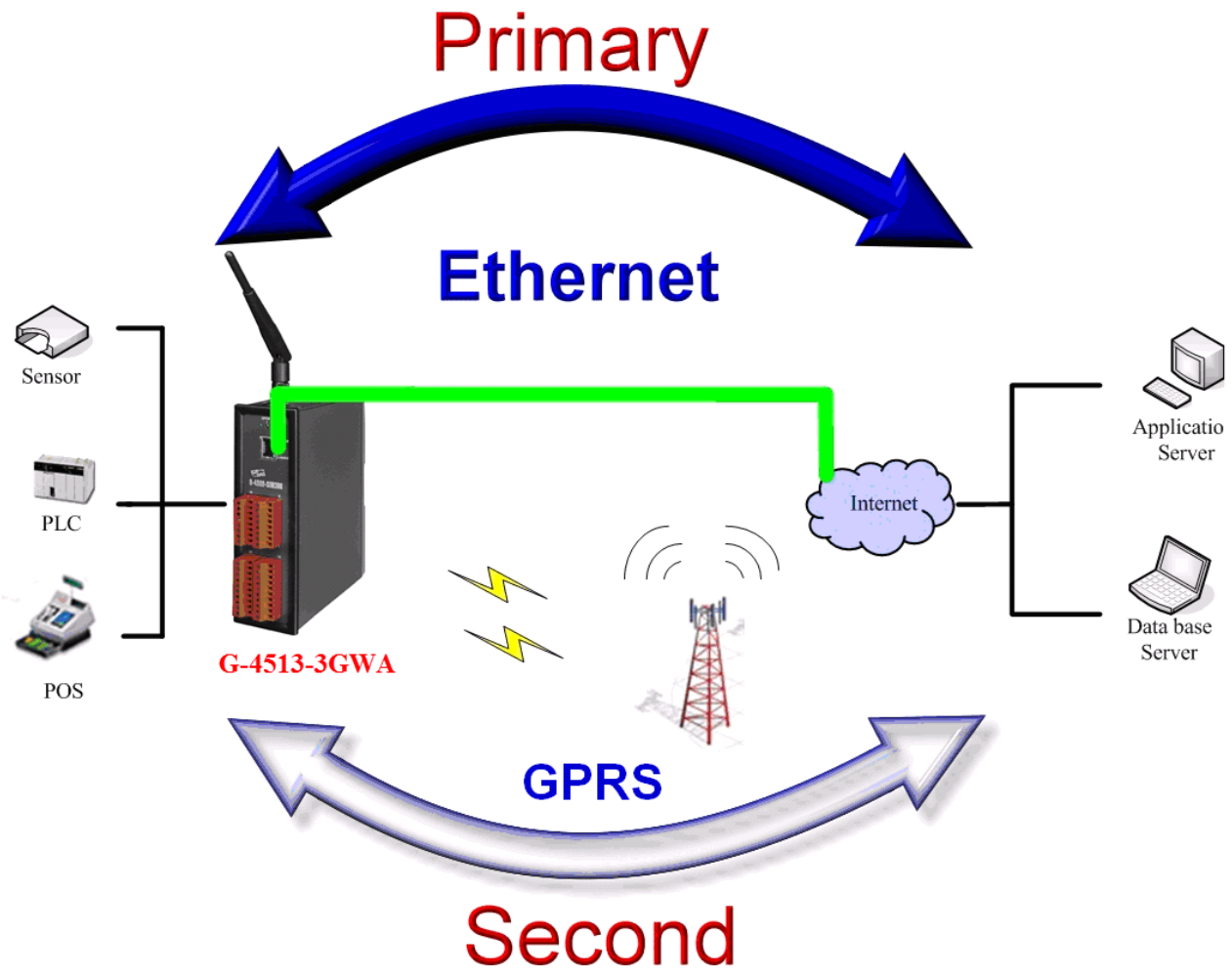
3. Application Architecture

3.1 Hydrologic / Wind Monitoring Application



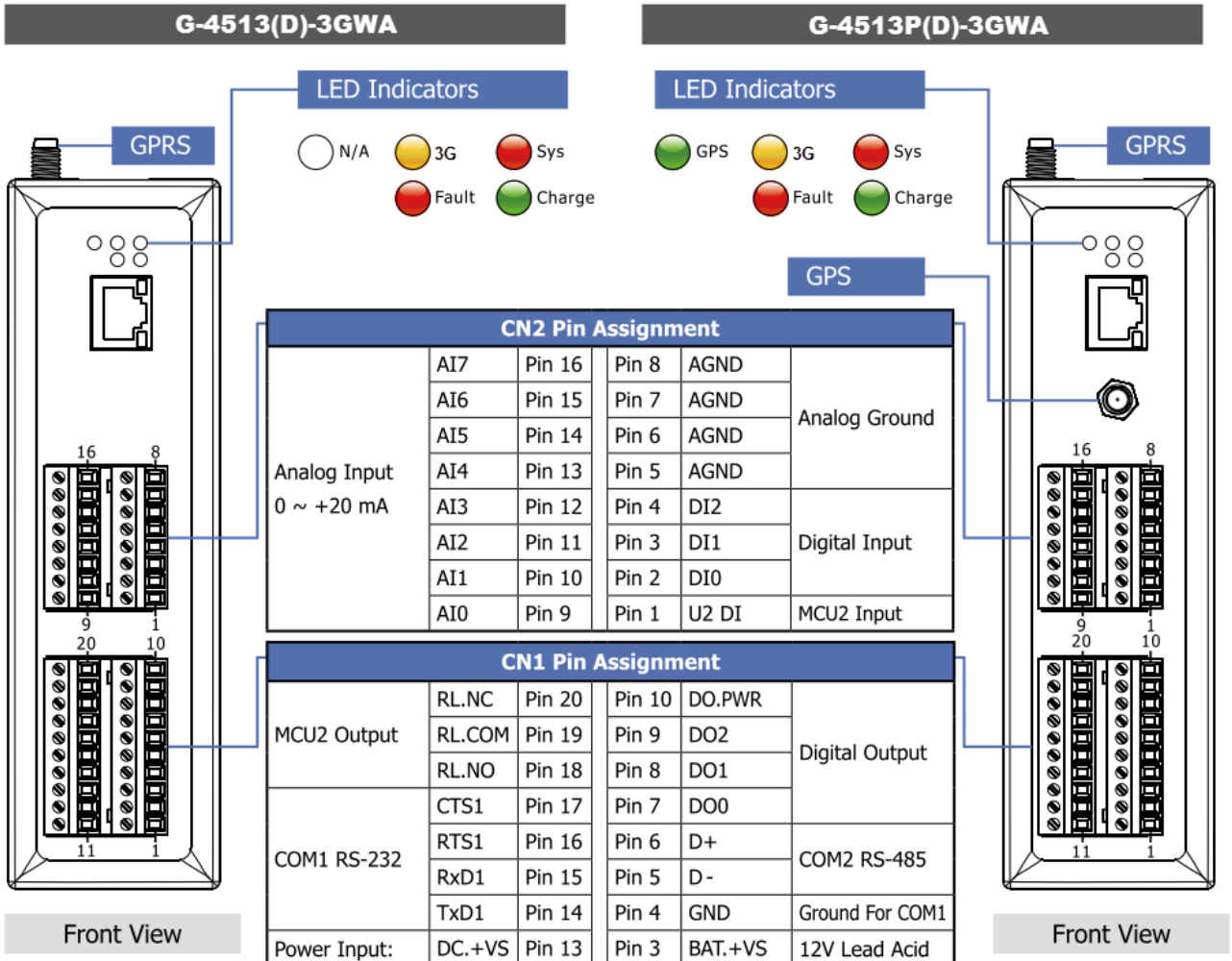
3.2 Car Monitor / Tracking System

3.3 Redundancy Communication system



4. Hardware

4.1 Pin Assignments

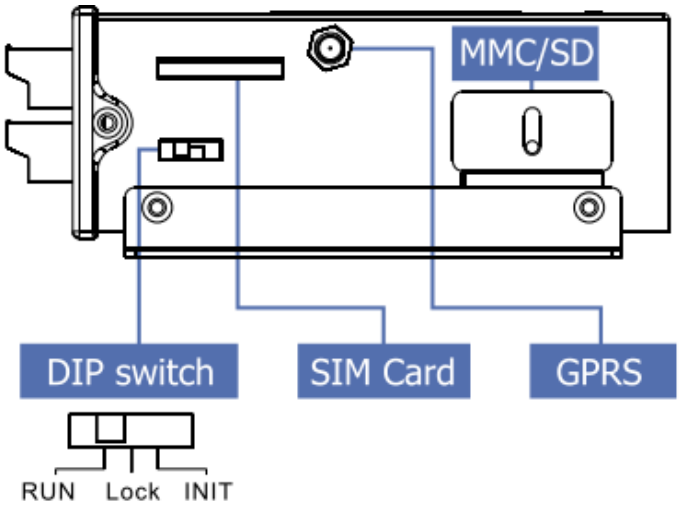


CN2 Pin Assignment

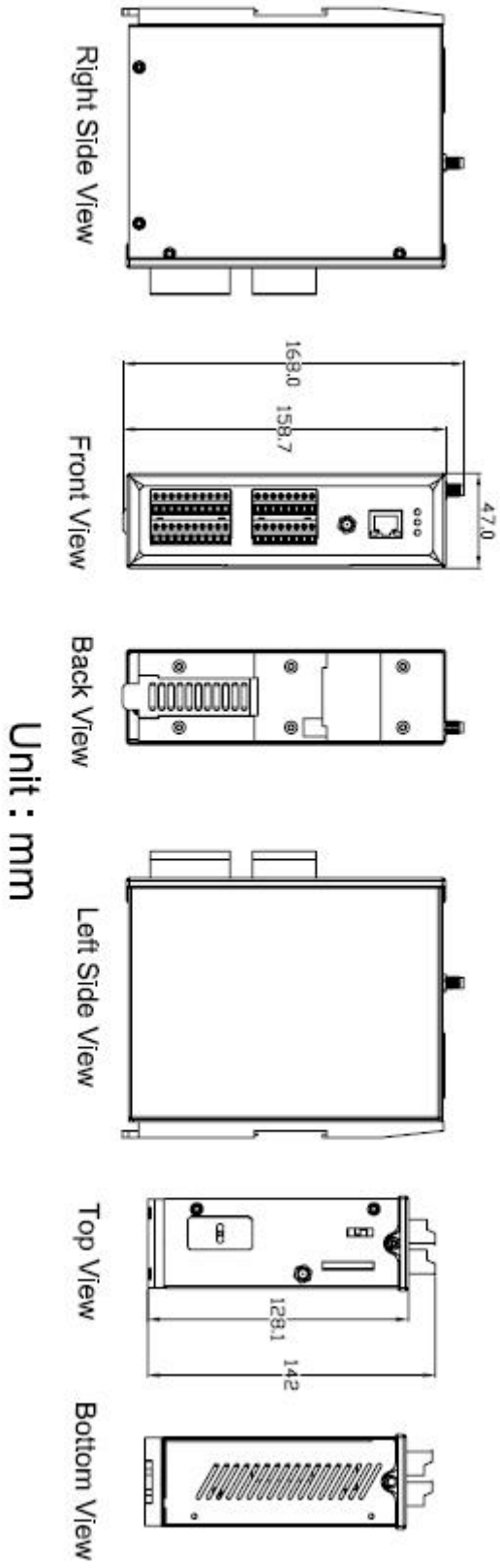
Analog Input 0 ~ +20 mA	AI7	Pin 16	Pin 8	AGND	Analog Ground
	AI6	Pin 15	Pin 7	AGND	
	AI5	Pin 14	Pin 6	AGND	
	AI4	Pin 13	Pin 5	AGND	
	AI3	Pin 12	Pin 4	DI2	Digital Input
	AI2	Pin 11	Pin 3	DI1	
	AI1	Pin 10	Pin 2	DI0	
	AI0	Pin 9	Pin 1	U2 DI	MCU2 Input

CN1 Pin Assignment

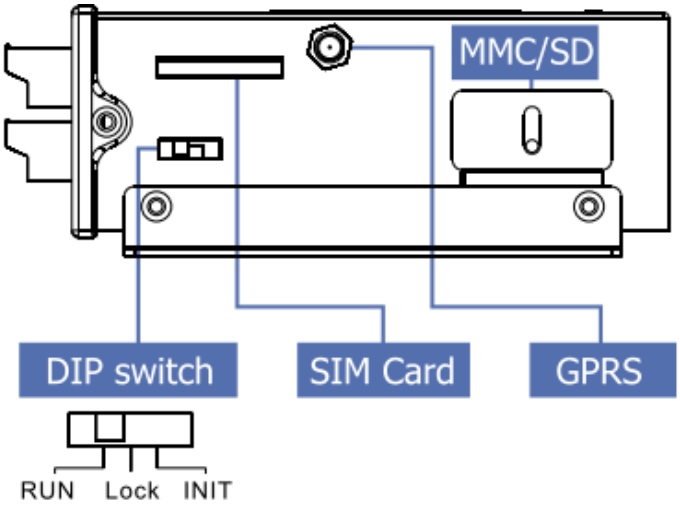
MCU2 Output	RL.NC	Pin 20	Pin 10	DO.PWR	Digital Output
	RL.COM	Pin 19	Pin 9	DO2	
	RL.NO	Pin 18	Pin 8	DO1	
COM1 RS-232	CTS1	Pin 17	Pin 7	DO0	
	RTS1	Pin 16	Pin 6	D+	COM2 RS-485
	RxD1	Pin 15	Pin 5	D-	
	TxD1	Pin 14	Pin 4	GND	Ground For COM1
Power Input: +10 ~ +30 VDC	DC.+VS	Pin 13	Pin 3	BAT.+VS	12V Lead Acid
	DC.GND	Pin 12	Pin 2	BAT.+GND	Battery
	N/A	Pin 11	Pin 1	F.G.	Frame Ground



4.2 Dimensions

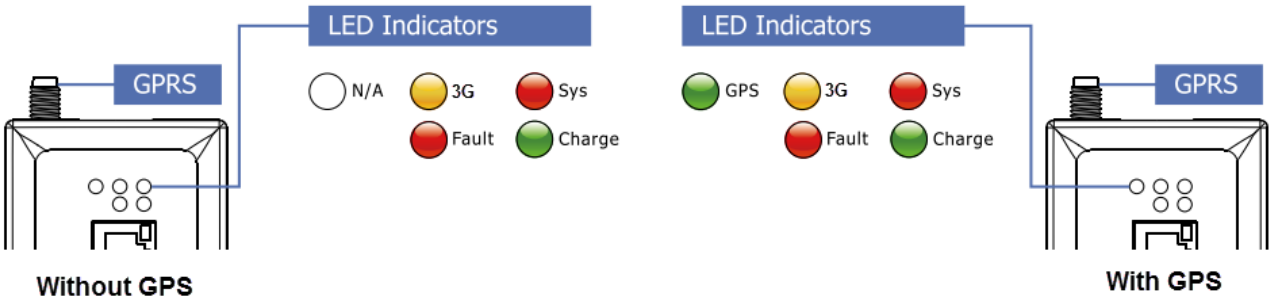


4.3 Operation Mode Switch



Operation Mode Switch	
RUN	OS can execute autoexec.bat
	Flash can be read/write.
LRun	OS can execute autoexec.bat
	Flash is read only (lock).
INIT	OS can not execute autoexec.bat
	Flash can be read/write.

4.4 LED Indicators



There are five LED indicators to help users to judge the various conditions of G-4513 Series. The description is as follows：

A. Sys (Red)：System LED is programmable.

B. 3G (Yellow)：The modem LED can indicate the status of 3G module.

	3G module normal	3G module fail
2G Mode	Blinking per 2 sec	OFF or Blinking (not 3 sec)
3G Mode	Blinking twice per 2 sec	

C. GPS (Green)(Option)：The GPS LED can indicate the status of GPS module.

GPS Fail	Search GPS	Receive GPS data
Always OFF	Always ON	Blinking (1 sec)

D. Charging (Green)：Charging status indicator.

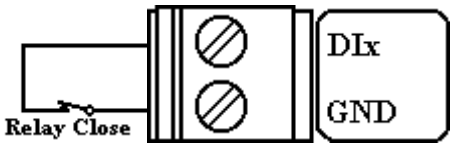
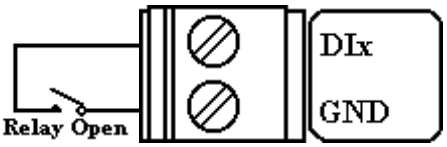
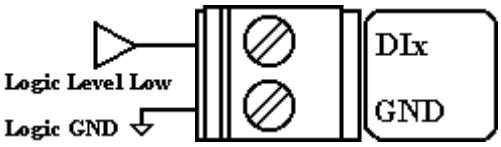
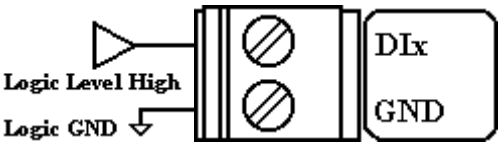
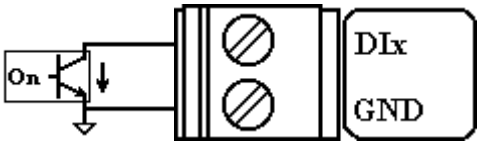
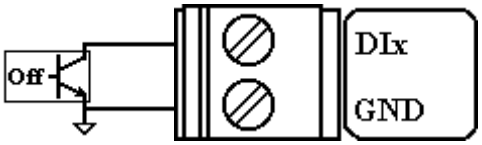
Charging	Not Charging
Always ON	Always OFF

E. Fault (Red)：Charging Fault indicator.

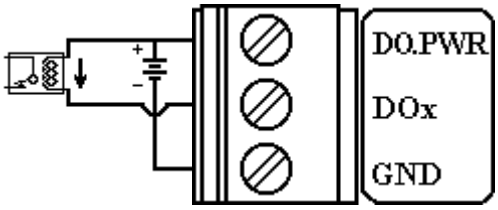
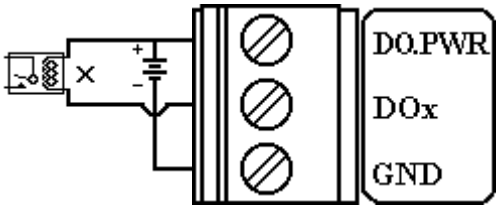
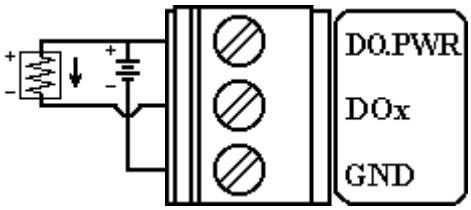
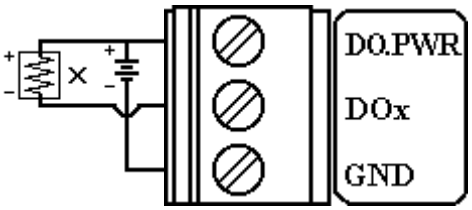
Normal	Fault
Always OFF	Always ON

4.5 Wire Connection

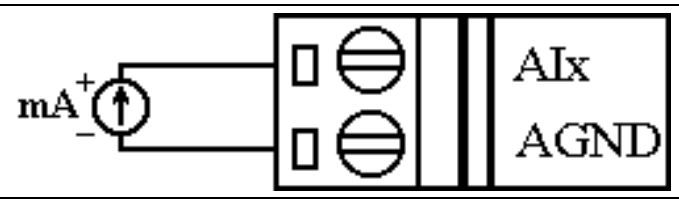
➢ Digital Input Wire Connection

Input Type	ON State DI value as 0	OFF State DI value as 1
Relay Contact		
TTL/CMOS Logic		
Open Collector		

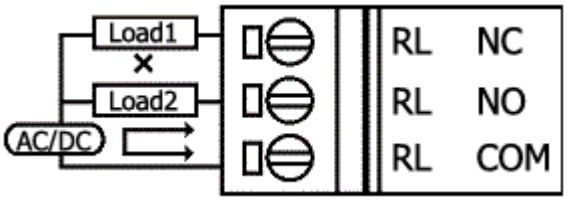
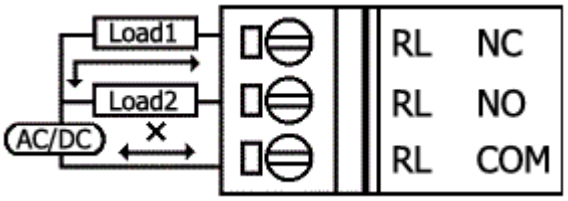
➢ Digital Output Wire Connection

Input Type	ON State DO value as 1	OFF State DO value as 0
Drive Relay		
Resistance Load		

- Current Input Wire Connection

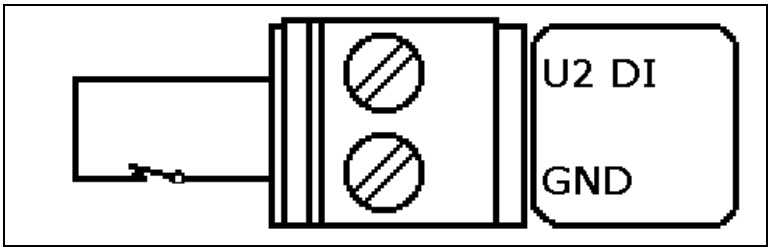
Input Type	

- Relay Wire connection

Relay Output ON	Relay Output OFF
	

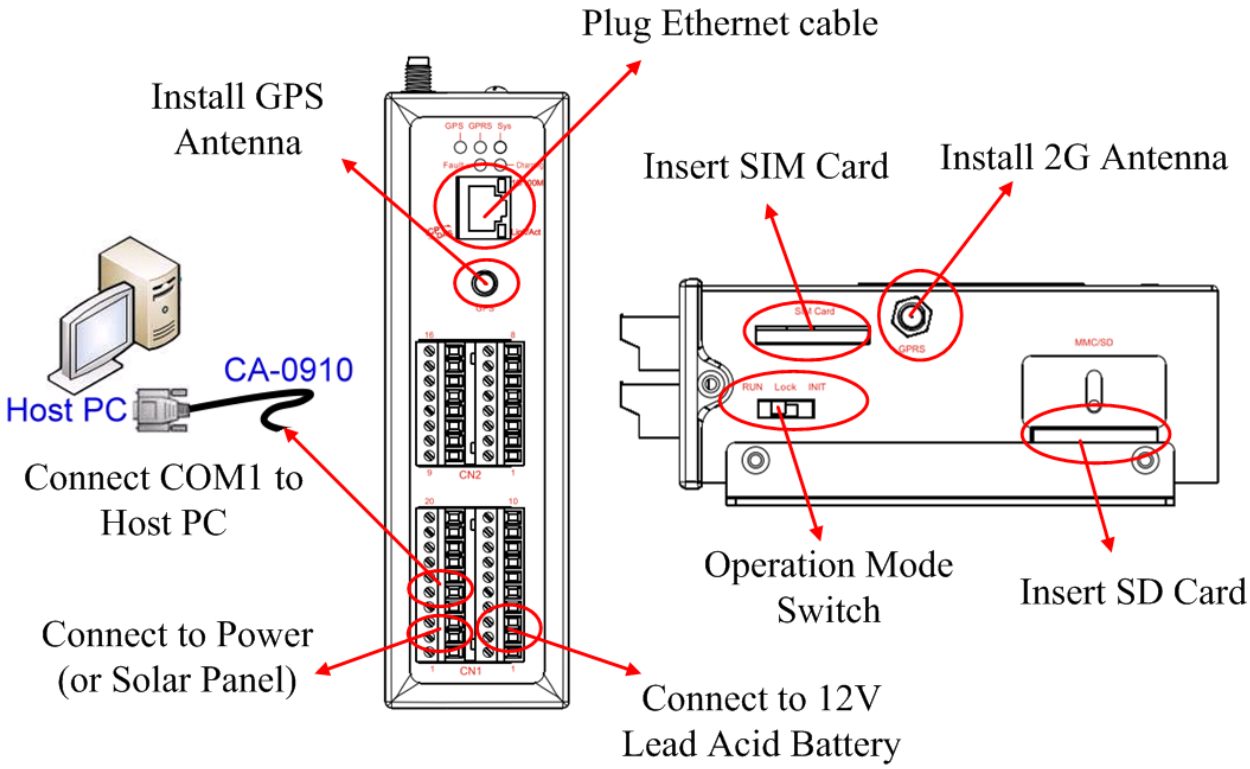
4.6 Wake up from Sleep Mode

1. When G-4513 was in sleep mode, you can connect the Pin "U2 DI" (MCU2 DI) to GND. to awaking G-4513 from sleep mode.
2. You will read "U2 DI" as 0, when you connect "U2 DI" to GND.



4.7 Installation

1. Install 2G/3G antenna.
2. Plug in the normal SIM card (Before apply the SIM card, confirm it is OK by mobile phone.)
3. Connect the DC.+VS and DC.GND to the power supply or Solar Panel.
4. Connect BAT.+VS and BAT.GND to the 12V Lead Acid Battery.



5. Power Saving and Charger

5.1 Power Saving

Sleep Mode

This mode will shut down 7186 CPU, all I/O(3DI, 3DO, 8AI, exclude MCU2 I/O) and GPS, but 3G module still works.

- Power Consumption：14~15 mA@12V
- How to awake G-4513：
  (1) Sleeping Time is finished.
  (2) Trigger U2_DI.(connect U2_DI to GND.; read U2_DI as 0)
  (3) Make a phone call to G-4513

Deep Sleep Mode

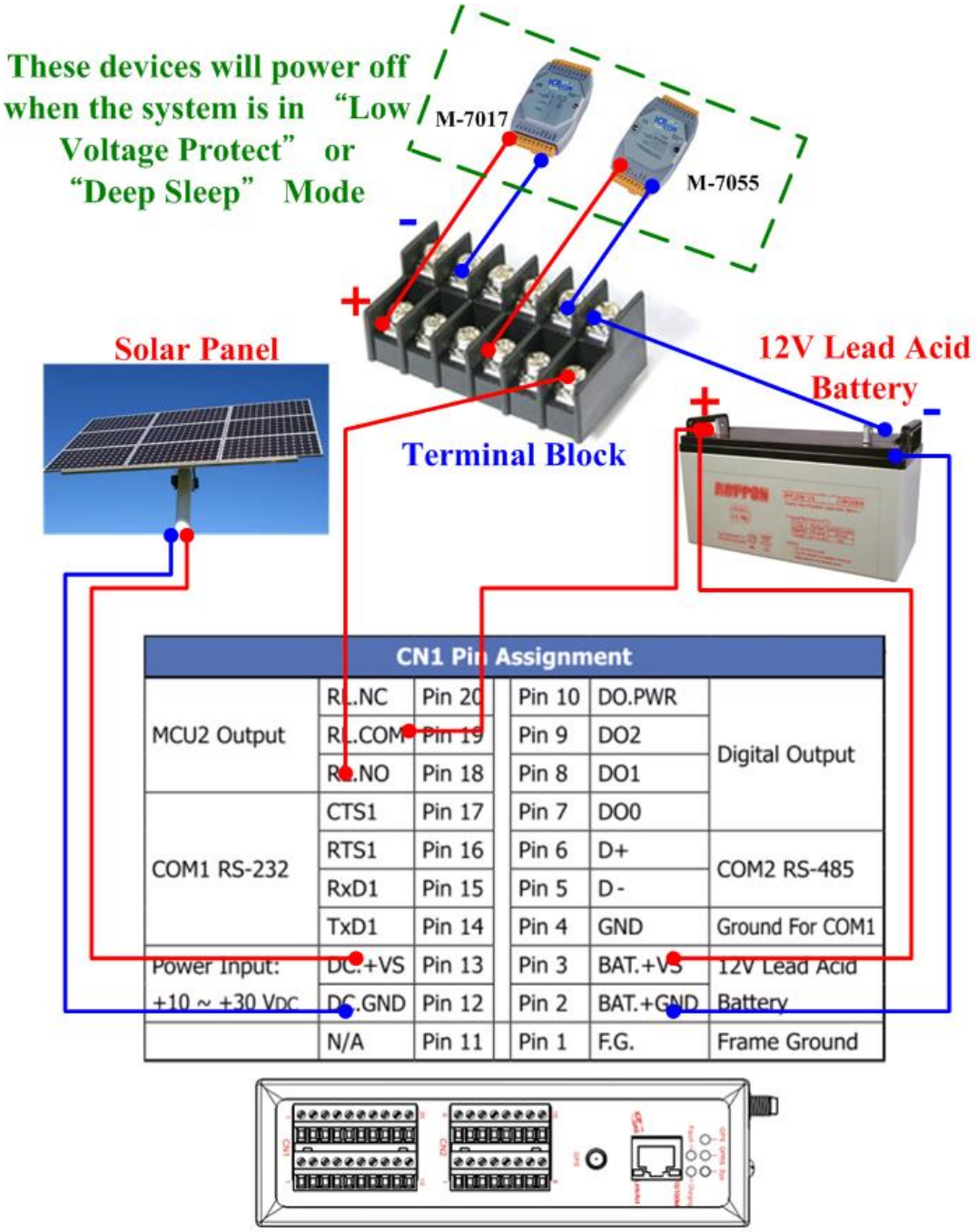
This mode will shut down all interface, but exclude MCU2 I/O.

- Power consumption：9~10 mA@12V
- How to awake G-4513：
  (1) Sleeping Time is finished.
  (2) Trigger U2_DI.(connect U2_DI to GND.; read U2_DI as 0)

Low Voltage Protection

Default value is disabled, and you can enable this function by your program. This function will prevent the battery to over-discharging. When the voltage of the battery is less 11.1V, G-4513 will go into Low Voltage Protect Mode that will turn off all system power. And then G-4513 will wake up if the voltage of the battery is over 12.6V after charging.

- Low Voltage disconnect Voltage = 11.1 V
- Low Voltage reconnect Voltage = 12.6 V
- How to use：please refer to the figure' s wire connection below, and MCU2 library demo code.

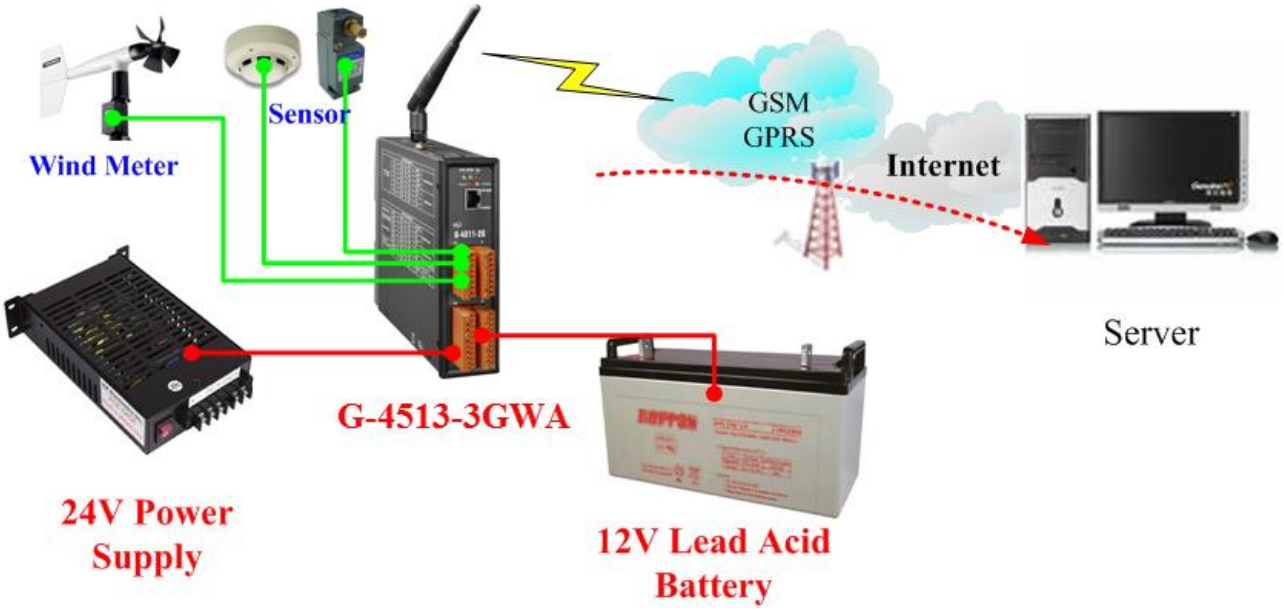

These devices will power off when the system is in "Low Voltage Protect" or "Deep Sleep" Mode
M-7017
M-7055
Solar Panel
Terminal Block
12V Lead Acid Battery

CN1 Pin Assignment					
MCU2 Output	RL.NC	Pin 20	Pin 10	DO.PWR	Digital Output
	RL.COM	Pin 19	Pin 9	DO2	
	RL.NO	Pin 18	Pin 8	DO1	
COM1 RS-232	CTS1	Pin 17	Pin 7	DO0	
	RTS1	Pin 16	Pin 6	D+	COM2 RS-485
	RxD1	Pin 15	Pin 5	D-	
	TxD1	Pin 14	Pin 4	GND	Ground For COM1
Power Input: +10 ~ +30 VDC	DC.+VS	Pin 13	Pin 3	BAT.+VS	12V Lead Acid Battery
	DC.GND	Pin 12	Pin 2	BAT.+GND	
	N/A	Pin 11	Pin 1	F.G.	Frame Ground

5.2 How to Choose the Battery

This section will discuss how to choose a suitable battery for your system. Because the alive time of the system is depending on your system power consumption and your battery capacity, we will calculate with some conditions below.

➢ **Example 1：**



Conditions：

- The system has a 24V external power
- It must work in 2 week when the external power is shut-down.
- The system transmits the data to the server every 10 minutes. (1 minute for full work, and 9 minutes for sleeping)
- Power consumption of deep sleep mode is 7.2 mA@12V
- Average power consumption of full work is 245 mA@12V

Calculation：

- Average power consumption = 245 x (1/10) + 7.2 x (9/10) = 31 (mA)
- 31 (mA) x 24 (hours) x 14 (days) = 10416 mAh
- We may choose "12V, 14Ah Lead Acid Battery" for this system.
- Because 10% battery capacity is low battery voltage state, we don't work in this state. We use 90% battery capacity to calculate.
- Double check the battery capacity：
  14Ah x 90% x 1000 = 12600 mAh > 10416 mAh
  → We will choose a **"12V, 14Ah Lead Acid Battery"** for this system.

Please refer Table 5.2.1 to choose the battery for other report frequency：

Table 5.2.1

Report Frequency	Average power consumption	power consumption of 14 days (mAh @12V)
Every minute (No Sleep)	245	82320
Every 10 minutes	31	10416
Every hour	11.2	3763.2
Every day	7.4	2486.4
Every Month	7.2	2419.2

➢ **Example 2：**

Conditions：

- The system has a 24V external power.
- It must work in 2 week when the external power is shut-down.
- The system transmits 3 Modbus devices data to the server every 10 minutes. (1 minute for full work, and 9 minutes for sleeping)
- The system will power off all Modbus devices by "MCU2 Relay Output" when it's in Deep Sleep Mode.
- Power consumption of deep sleep mode is 7.2 mA@12V
- Average power consumption of full work is 424 mA@12V

Calculation：

- Average power consumption = 424 x (1/10) + 7.2 x (9/10) = 49 (mA)
- 424 (mA) x 24 (hours) x 14 (days) = 16430.4 mAh
- We may choose "12V, 22Ah Lead Acid Battery" for this system.
- Because 10% battery capacity is low battery voltage state, we don't work in this state. We use 90% battery capacity to calculate.
- Double check the battery capacity:
  22Ah x 90% x 1000 = 19800 mAh > 16430.4 mAh
  → We will choose a **"12V, 22Ah Lead Acid Battery"** for this system.

Please refer Table 5.2.2 to choose the battery for other report frequency：

Table 5.2.2

Report Frequency	Average power consumption	power consumption of 14 days (mAh @12V)
Every minute (No Sleep)	424	142464
Every 10 minutes	48.9	16430.4
Every hour	14.1	4737.6
Every day	7.5	2520
Every Month	7.2	2419.2

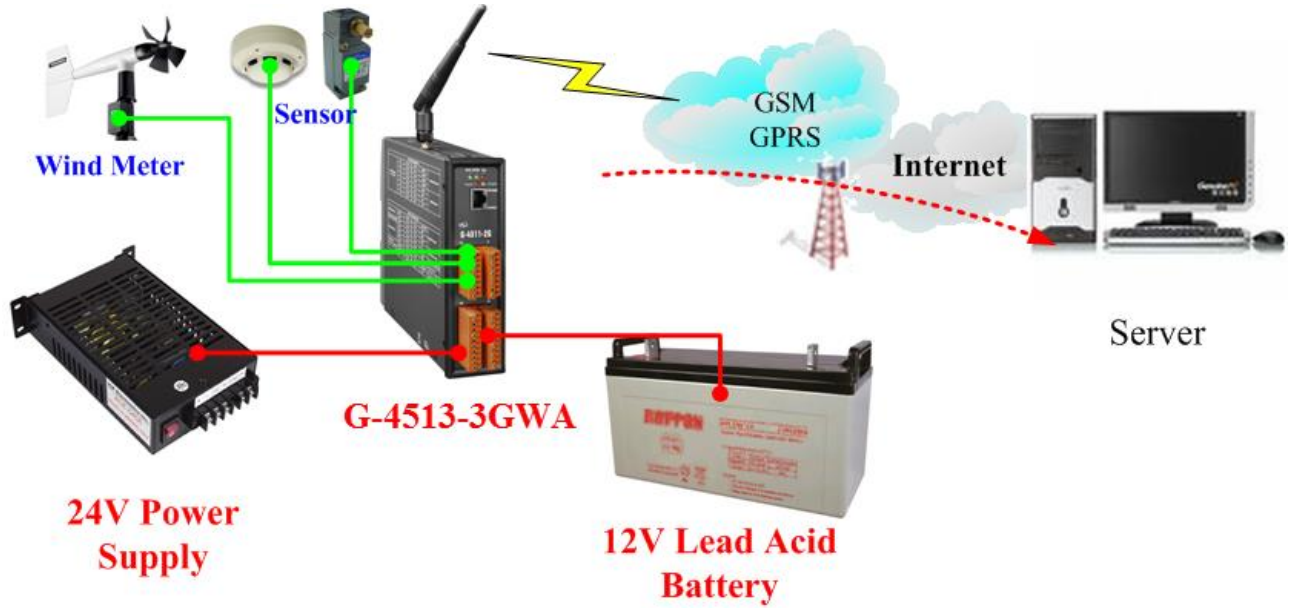
5.3 How to Choose the Solar Panel

This section will discuss how to choose a suitable solar panel for your system. The power of solar panel must be more than the power consumption of the system, and we will calculate with some conditions below.

Things you must know before you calculate

Charging voltage：must be more than +16V
Max. Charge Current：2A
Sun hours：an average value. If the average daily solar radiation of the area is 3 kW/m2 , the sun hours of this area is 3 hours

➢ **Example：**



Conditions：

- Power consumption of deep sleep mode is 7.2 mA@12V
- Average power consumption of the system is 245 mA@12V
- The system transmits the data to the server every 10 minutes. (1 minute for full work, and 9 minutes for sleeping)
- Sun hours is 4 hours/day.
- Using 10W solar panel

Calculation：

- Solar panel Max. current = 10 (w) / 12 (V) = 0.833 (A) = 833 (mA)
- Usually the charging current may be effected by many factor, like angle, building and other environment factor…etc.
- Here we use 1/2 Max. current to calculate
- 1/2 Max. Current = 833/2 = 416 mA
- Average current = 416 x 4 / 24 = 69 (mA/hr)
- Refer to Table 5.3.2, We can know "Average power consumption" is 31 mA for this system. (refer to Table 5.3.2 or section 5.2)

  → We can know **10W solar Panel is suitable** for this system, because 69 > 31

Table 5.3.1

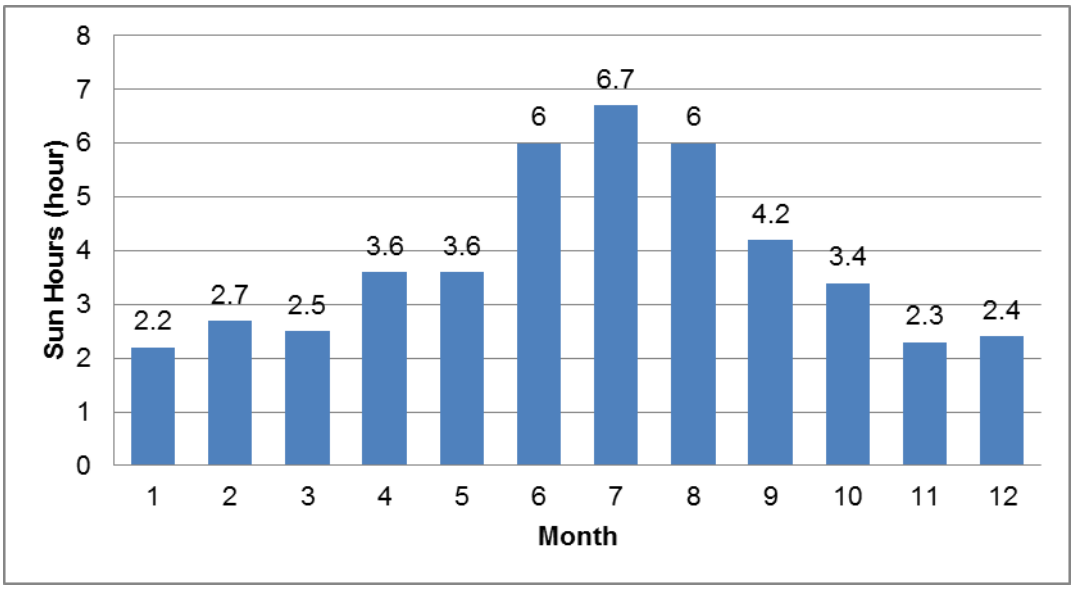
Solar panel (W)	Max. Current (mA)	1/2 Current (mA)	Sun hours	Average Current for a day (mA/hr)
10	833	416	4	69
20	1666	833	4	138
30	2500	1250	4	208
40	3333	1666	4	277
50	4166	2083	4	347

Table 5.3.2

Report Frequency	Average power consumption	power consumption of 14 days (mAh @12V)
Every minute (No Sleep)	245	82320
Every 10 minutes	31	10416
Every hour	11.2	3763.2
Every day	7.4	2486.4
Every Month	7.2	2419.2

Things you must know about "sun hours"

Usually the "sun hours" is variable with the season. You need use the Min. sun hours to calculate and choose the solar panel and the battery.
For example, if the sun hours of month is like the figure below, you must use 2.2 hours to calculate but not 6.7 hours.



6. Function Introduction

6.1 I/O Library Function Definition and Description

Function definition	Description
X305IO_Init	Initial I/O
X305IO_GetLibVersion	Get X305IO_LIB Version
X305IO_Read_AD_CalibrationGain	Read AD Calibration Gain
X305IO_Read_AD_CalibrationOffset	Read AD Calibration Offset
X305IO_AnalogIn	Read value from assign AI channel
X305IO_Read_All_DI	Read All DI
X305IO_Read_One_DI	Read the value form assign DI channel
X305IO_Write_All_DO	Write All DO
X305IO_Write_One_DO	Write the value to the assign DO channel
X305IO_Read_All_DO	Read All DO state
X305IO_Read_One_DO	Read the DO state form the assign DO channel.
X305IO_AnalogIn_SetChannel	Set the AI channel that users want to read.
X305IO_AnalogIn_Hex	Read the value from the specific A/D channel (12 bits)
X305IO_AnalogIn_HexToFloat	Transfer the AI value from 12 bits to float

6.1.1 X305IO_Init

Initial X305IO.

Syntax

```
int X305IO_Init(void);
```

Parameters

None

Return values

0：success
<>0：error

6.1.2 X305IO_GetLibVersion

Get X305IO_Lib Version.

Syntax

```
unsigned X305IO_GetLibVersion(void);
```

Parameters

None

Return values

Version Number

6.1.3 X305IO_Read_AD_CalibrationGain

Read the A/D Calibration Gain.

Syntax

```
float X305IO_Read_AD_CalibrationGain(void);
```

Parameters

None

Return values

Calibration Gain of the AD channels

6.1.4 X305IO_Read_AD_CalibrationOffset

Read the A/D Calibration Offset.

Syntax

```
float X305IO_Read_AD_CalibrationOffset(void);
```

Parameters

None

Return values

Calibration Offset of the AD channels

6.1.5 X305IO_AnalogIn

Read the value from the assign AI channel.

Syntax

```
float X305IO_AnalogIn(
    int iChannel
);
```

Parameters

iChannel

0：channel 0
1：channel 1
2：channel 2
3：channel 3
4：channel 4
5：channel 5
6：channel 6
7：channel 7

Return values

0.0mA ~ 20.0mA

6.1.6 X305IO_Read_All_DI

Read all DI values of the G-4513 series.

Syntax

```
int X305IO_Read_All_DI(void);
```

Parameters

None

Return values

0x00~0x07

Example

When DI0 Ground
DI1 Open
DI2 Open

value = X305IO_Read_All_DI( );

value = 0x6

6.1.7 X305IO_Read_One_DI

Read the value from the assign DI channel.

Syntax

```
int X305IO_Read_One_DI(
    int iChannel
);
```

Parameters

iChannel

0：channel 0

1：channel 1

2：channel 2

Return values

1：open

Logic high level (+3.5V ~ +30V)

0：close to GND

Logic low level (0V ~ +1V)

6.1.8 X305IO_Write_All_DO

Write to all DO values of the G-4513 series.

Syntax

```
void X305IO_Write_All_DO(
    int iOutValue
);
```

Parameters

iOutValue
0x0~0x7

Return values

None

Example

X305IO_Write_All_DO(6);

After function execute：
DO0 OFF
DO1 ON
DO2 ON

6.1.9 X305IO_Write_One_DO

Write the specific value to the assign DO channel.

Syntax

```
void X305IO_Write_One_DO(
    int iChannel, int iStatus
);
```

Parameters

iChannel

0：channel 0
1：channel 1
2：channel 2

iStatus

0：Status is OFF
1：Status is ON

Return values

None

6.1.10 X305IO_Read_All_DO

Read all DO values of the G-4513 series.

Syntax

```
int X305IO_Read_All_DO(void);
```

Parameters

None

Return values

0x0~0x7

Example

When DO0 OFF
DO1 ON
DO2 ON

Value = X305IO_Read_All_DO( );

Value = 0x6

6.1.11 X305IO_Read_One_DO

Read the state from the assign DO channel.

Syntax

```
int X305IO_Read_One_DO(
    int iChannel
);
```

Parameters

iChannel

0：channel 0

1：channel 1

2：channel 2

Return values

0：OFF

1：ON

6.1.12 X305IO_AnalogIn_SetChannel

Set the specific AI channel that users want to read.

Syntax

```
int X305IO_AnalogIn_SetChannel(
    unsigned iChannel
);
```

Parameters

iChannel

0：channel 0
1：channel 1
2：channel 2
3：channel 3
4：channel 4
5：channel 5
6：channel 6
7：channel 7

Return values

0：Set up success
-1：Set iChannel number error

6.1.13 X305IO_AnalogIn_Hex

Read the value of the assign AI channel assigned by X305IO_AnalogIn_SetChannel function.

Syntax

```
int X305IO_AnalogIn_Hex(void);
```

Parameters

None

Return values

After Read assign AI channel value.

Example

```
X305IO_AnalogIn_SetChannel(0);      // Set channel 0
X305IO_AnalogIn_Hex( );
```

6.1.14 X305IO_AnalogIn_HexToFloat

Set the AI value from 12 bits to float format.

Syntax

```
float X305IO_AnalogIn_HexToFloat(
    int iValue
);
```

Parameters

iValue

A value want to 12 bits transform float.

Return values

The transferred AI value by float format.

Example

Set the channel 0 to read, and then transform the value to float.

```
float AdValue;

X305IO_AnalogIn_SetChannel(0);
AdValue=X305IO_AnalogIn_HexToFloat(X305IO_AnalogIn_Hex( ));
```

6.2 MMC/SD Library Function Definition and Description

Function definition	Description
PC_Ertfs_Init	Configure ERTFS drive letter/device mapping and initialize device drivers.
PC_Open	Open a file
PC_Read	Read bytes from the file
PC_Write	Write Bytes to the file
PC_Close	Close the file and flush the file allocation table
Get_ErrNo	Get Error code
PC_lseek	Move file pointer
PC_MKDir	Create a subdirectory
PC_RMDir	Delete a directory
PC_deltree	Delete a directory tree
PC_MV	Rename a file or directory
PC_IsDir	Test if a path is a directory
PC_Pwd	Return the current working directory
PC_Get_Attributes	Get File Attributes
PC_Set_Attributes	Set File Attributes

Tips & Warnings

Before using these functions, users must format the MMC/SD card as FAT16 from the PC.

6.2.1 PC_Ertfs_Init

Configure ERTFS drive letter/device mapping and initialize device drivers.

Syntax

```
BOOLEAN pc_ertfs_init(void);
```

Parameters

None

Return values

True：success
Fail：no success

6.2.2 PC_Open

Open/Create the file from MMC/SD card with the specific mode.

Syntax

```
PCFD pc_open(
    char *name,
    word flag,
    word mode
);
```

Parameters

name

The file path in MMC/SD card

flag

PO_APPEND：Seek to eof on each write.
PO_BINARY：Ignored. All file access is binary.
PO_TEXT：Ignored.
PO_RDONLY：Open for read only
PO_RDWR：Read/write access allowed.
PO_WRONLY：Open for write only.
PO_CREAT：Create the file if it does not exist.
PO_EXCL：If flag has (PO_CREAT | PO_EXCL) and the file already exists, fail and set get_errno() to PEEXIST.
PO_TRUNC：Truncate the file if it already exists.
PO_NOSHAREANY：Fail if the file is already open.
PO_NOSHAREWRITE：Fail if the file is already open for write.

mode

PS_IWRITE：Write permitted.
PS_IREAD：Read permitted. (Always true anyway)

Return values

Returns a non-negative integer to be used as a file descriptor for calling po_read, po_write, po_lseek, po_flush, po_truncate, and po_close, otherwise it returns -1 and get_errno.

Example

This example below would show to open test1 file and set the file allowing read or write.

```
char *fname="test1\\test1.txt";

pc_ertfs_init();
pc_open(fname,
(word) (PO_BINARY|PO_RDWR|PO_CREAT|PO_APPEND),
(word) (PS_IWRITE | PS_IREAD) );
```

6.2.3 PC_Read

Read bytes from a file.

Syntax

```
int pc_read(
    PCFD fd,
    byte *buf
    word count
);
```

Parameters

fd

file desc

**buf*

data buf

count

data length

Return values

Returns the actual number of bytes read or Oxffff on error. If the return value is Oxffff, get_errno will return one of the following：

PEBADF：Invalid file descriptor
PENOSPC：Write failed. Presumably because of no space

Example

```
char *fname="test1\\test1.txt";
PCFD    out_fd;
char bff[129];

pc_ertfs_init();
out_fd = pc_open(fname,
(word) (PO_BINARY|PO_RDWR|PO_CREAT|PO_APPEND),
(word) (PS_IWRITE | PS_IREAD) );

pc_read(out_fd, bff, 512);
```

6.2.4 PC_Write

Write Bytes to a file.

Syntax

```
int pc_write(
    PCFD fd,
    byte *buf,
    word count
);
```

Parameters

fd

file desc

**buf*

write data buf

count

data length

Return values

Returns the actual number of bytes read or Oxffff on error. If the return value is Oxffff, get_errno will return one of the following：

PEBADF：Invalid file descriptor

PENOSPC：Write failed. Presumably because of no space

Example

This example will open test1 file, and set read/write is allowed.And write a data.

```
char *fname="test1\\test1.txt";
PCFD    out_fd;
char bf[129],bff[129];
long testcnt=0;
int len;

pc_ertfs_init();
out_fd = pc_open(fname,
(word) (PO_BINARY|PO_RDWR|PO_CREAT|PO_APPEND),
(word) (PS_IWRITE | PS_IREAD) );

len=sprintf(bf,"Line:%09lu\r\n",testcnt++);
pc_write(out_fd,bf,len);
```

6.2.5 PC_Close

Close a file.

Syntax

```
int pc_close(
    PCFD fd
);
```

Parameters

fd

file desc

Return values

Returns 0 if all went well, otherwise -1. If -1 is returned, get_errno will return one of these values：

PEBADF：Invalid file descriptor
PENOSPC：Write failed. Presumably because of no space

6.2.6 Get_ErrNo

Get error code.

Syntax

```
int get_errno(void);
```

Parameters

None

Return values

2：PENOENT - File not found or path to file not found.
9：PEBADF - Invalid file descriptor.
13：PEACCESS - Attempt to open a read only file or a special (directory).
17：PEEXIST - Exclusive access requested but file already exists.
22：PEINVAL - Seek to negative file pointer attempted.
24：PEMFILE - No file descriptors available (too many files open).
28：PENOSPC - Write failed. Presumably because of no space.
30：PESHARE - Open failed do to sharing.
31：PEDVICE - No Valid Disk Present.
32：PEBADDIR - DELTREE -- Directory structure corrupt.

6.2.7 PC_lseek

Move file pointer.

Syntax

```
long pc_lseek(
    PCFD fd,
    long offset,
    int origin
);
```

Parameters

fd

file desc

offset

offset value

origin

PSEEK_SET：offset from begining of file
PSEEK_CUR：offset from current file pointer
PSEEK_END：offset from end of file

Return values

If success, Returns new offset value, otherwise -1. If -1 is returned, get_errno will return one of these values：

PEBADF：Invalid file descriptor
PEINVAL：Seek to negative file pointer attempted

6.2.8 PC_MKDir

Create a subdirectory.

Syntax

```
BOOLEAN pc_mkdir(
    char *name
);
```

Parameters

name

Name of directory to be created.

Return values

Returns TRUE if the subdirectory was created, otherwise FALSE.If FALSE is returned, get_errno will return one of these values：

PENOENT：Directory not found
PEEXIST：File or directory already exists
PENOSPC：Write failed

Example

```
pc_mkdir("USR\\LIB");
```

6.2.9 PC_RMDir

Delete the directory specified in path. Fails if path is not a directory, is read only or is not empty.

Syntax

```
BOOLEAN pc_rmdir(
    char    *name
);
```

Parameters

name

Name of directory to be deleted.

Return values

TRUE if the directory was successfully removed, otherwise FALSE. If FALSE is returned, get_errno will return one of these values：

PENOENT：Directory not found
PEACCES：Not a directory, not empty or in use
PENOSPC：Write failed

6.2.10 PC_deltree

Delete the directory specified in name, all subdirectories of that directory, and all files contained therein. Fail if name is not a directory, is read only or is currently in use.

Syntax

```
BOOLEAN pc_deltree(
    char    *name
);
```

Parameters

name

Name of directory tree to be deleted.

Return values

Returns TRUE if the directory was successfully removed. If FALSE is returned, get_errno will return one of these values：

PENOENT：Directory not found or path to file not found
PEACCES：Not a directory, not empty or in use
PENOSPC：Write failed

6.2.11 PC_MV

Renames the file oldpath to newname. Fails if newname is invalid, already exists or oldpath is not found.

Syntax

```
BOOLEAN pc_mv(
    char *name,
    char *newname
);
```

Parameters

name
the file oldpath

newname
New the file name

Return values

Returns TRUE if the file was renamed, otherwise FALSE.If FALSE is returned, get_errno will return one of these values：

PENOENT：Directory not found
PEEXIST：File or directory already exists
PENOSPC：Write failed

Example

```
if (!pc_mv("TEXT\\LETTER.TXT","TEXT\\NEWLETTER.TXT"))
    Print("Can't rename LETTER.TXT\n");
```

6.2.12 PC_IsDir

Test if a path is a directory.

Syntax

```
BOOLEAN pc_isdir(
    char *path
);
```

Parameters

path

The file path in MMC/SD card

Return values

Returns TRUE if path points to a valid existing directory, otherwise FALSE.

6.2.13 PC_Pwd

Return the current working directory.

Syntax

```
BOOLEAN pc_pwd(
    char *drive,
    char *path
);
```

Parameters

drive

""

path

return the current working directory.

Return values

Returns TRUE if a valid path was returned in path, otherwise no if the current working directory could not be found.

Tips & Warnings

Return buffer must contain enough space to hold the full path.

6.2.14 PC_Get_Attributes

Get File Attributes. Give a file name. Return the directory entry attributes associated with the entry. One or more of the following values will be or'ed together：

BIT	Nemonic
0	ARDONLY
1	AHIDDEN
2	ASYSTEM
3	AVOLUME
4	ADIRENT
5	ARCHIVE

Syntax

```
BOOLEAN pc_get_attributes(
    char *path,
    byte *p_return
);
```

Parameters

path

The file path in MMC/SD card

p_return

Return the directory entry attributes.

Return values

Returns TRUE if successful, otherwise it returns FALSE and get_errno returns one of these values：

PENOENT：Directory not found
PEEXIST：File or directory already exists
PENOSPC：Write failed

Example

```
byte attribs;

if(pc_get_attributes("test\\test1.txt", &attribs) {
    if(attribs & ARDONLY)
        Print("File is ARDONLY");
    if(attribs & AHIDDEN)
        Print("File is AHIDDEN");
    if(attribs & ASYSTEM)
        Print("File is ASYSTEM");
    if(attribs & AVOLUME)
        Print("File is AVOLUME");
    if(attribs & ADIRENT)
        Print("File is ADIRENT");
    if(attribs & ARCHIVE)
        Print("File is ARCHIVE");
    if(attribs & ANORMAL)
        Print("File is ANORMAL")
}
```

6.2.15 PC_Set_Attributes

Set File Attributes. Given a file or directory name set the directory entry attributes associated with the entry. One or more of the following values may be or'ed together：

BIT	Nemonic
0	ARDONLY
1	AHIDDEN
2	ASYSTEM
5	ARCHIVE

Syntax

```
BOOLEAN pc_set_attributes(
    char *path,
    byte attributes
);
```

Parameters

path

The file path in MMC/SD card

attributes

Set the directory entry attributes.

Return values

Returns TRUE if successful, otherwise FALSE and get_errno will return one of these values：

PENOENT：Couldn't find the entry

PENOSPC：Write failed

Example

```
byte attribute;
char *fname="test1\\test1.txt";
attribute= ARDONLY | AHIDDEN;

pc_set_attributes(fname,attribute);
```

6.3 LCD Library Function Definition and Description

Function definition	Description
LCD_Init	Initialize the library
LCD_BackLight_On	Turn on the LCD backlight
LCD_BackLight_Off	Turn off the LCD backlight
LCD_ShowText	Display one character on the LCD panel
LCD_DisplayNumber	Display number on the the LCD panel
LCD_SetNumber	Display one number on the specified position
LCD_ClrNumber	Clear the displayed number by one character position
LCD_ClrScrn	Clear the LCD panel
LCD_StandByMode	Enter the stand by mode
LCD_NormalMode	Restore the LCD to normal mode
LCD_GotoPosition	Move the cursor to the specified position
LCD_CursorDisplay	Set the Cursor display status
LCD_LineReverse	Select one of four line and reverse the display
LCD_LineRestore	Select one of four line and restore the display
LCD_GetLibDate	Gets the create date of funciton library
LCD_GetLibVersion	Gets the version number of function library

6.3.1 LCD_Init

Initialize parameters about LCD functions in the library.

Syntax

```
void LCD_Init(void);
```

Parameters

None

Return values

None

6.3.2 LCD_BackLight_On

Turn on the LCD backlight.

Syntax

```
void LCD_BackLight_On(void);
```

Parameters

None

Return

None

6.3.3 LCD_BackLight_Off

Turn off the LCD backlight.

Syntax

```
void LCD_BackLight_Off(void);
```

Parameters

None

Return values

None

6.3.4 LCD_ShowText

Display one character on the LCD panel, and the cursor will right-shifted by one character position automatically.

Syntax

```
void LCD_ShowText(
    uchar Text
);
```

Parameters

Text

Display character

Return values

None

6.3.5 LCD_DisplayNumber

After calling either the LCD_SetNumber or LCD_ClrNumber, it is necessary to call LCD_DisplayNumber to display number on the the LCD panel.

Syntax

```
void LCD_DisplayNumber(void);
```

Parameters

None

Return values

None

6.3.6 LCD_SetNumber

Display one number on the specified position.

Syntax

```
void LCD_SetNumber(
    int Line,
    int Offset,
    int Number
);
```

Parameters

Line

One of two line numbers (1 to 2)

Offset

Cursor position (1 to 5)

Number

Display number

Return values

None

Example

```
LCD_SetNumber(1, 1, 0);
LCD_SetNumber(1, 2, 1);
LCD_DisplayNumber();
```

6.3.7 LCD_ClrNumber

Clear the displayed number by one character position.

Syntax

```
void LCD_ClrNumber(
    int Line,
    int Offset
);
```

Parameters

Line

One of two line numbers (1 to 2)

Offset

Cursor position (1 to 5)

Return values

None

Example

```
LCD_ClrNumber(2, 3);
LCD_DisplayNumber();
```

6.3.8 LCD_ClrScrn

Clear the LCD panel.

Syntax

```
void LCD_ClrScrn(void);
```

Parameters

None

Return values

None

6.3.9 LCD_StandByMode

Enter the stand by mode, and it can be terminated by either LCD_NormalMode() or other function.

Syntax

```
void LCD_StandByMode(void);
```

Parameters

None

Return values

None

6.3.10 LCD_NormalMode

Restore the LCD to normal mode when it is in the stand by mode.

Syntax

```
void LCD_NormalMode(void);
```

Parameters

None

Return values

None

6.3.11 LCD_GotoPosition

Move the cursor to the specified position.

Syntax

```
void LCD_GotoPosition(
    int Line,
    int Offset
);
```

Parameters

Line

One of four line numbers (1 to 4)

Offset

Cursor position (1 to 8)

Return values

None

6.3.12 LCD_CursorDisplay

Set the Cursor display status.

Syntax

```
void LCD_CursorDisplay(
    int Display,
    int Blink
);
```

Parameters

Display

Cursor display on/off

1: Display on

0: Display off

Blink

Character blink on/off

1: Display on

0: Display off

Return values

None

6.3.13 LCD_LineReverse

Select one of four line and reverse the display.

Syntax

```
void LCD_LineReverse(int Line);
```

Parameters

Line

One of four line numbers (0 to 4)

Return values

None

6.3.14 LCD_LineRestore

Select one of four line and restore the display.

Syntax

```
void LCD_LineRestore(
    int Line
);
```

Parameters

Line

One of four line numbers (0 to 4)

Return values

None

6.3.15 LCD_GetLibDate

Gets the create date of funciton library.

Syntax

```
void LCD_GetLibDate(
    unsigned char *LibDate
);
```

Parameters

LibDate

Gets the create date of funciton library

Return values

None

6.3.16 LCD_GetLibVersion

Get the version number of function library.

Syntax

```
unsigned LCD_GetLibVersion(void);
```

Parameters

None

Return values

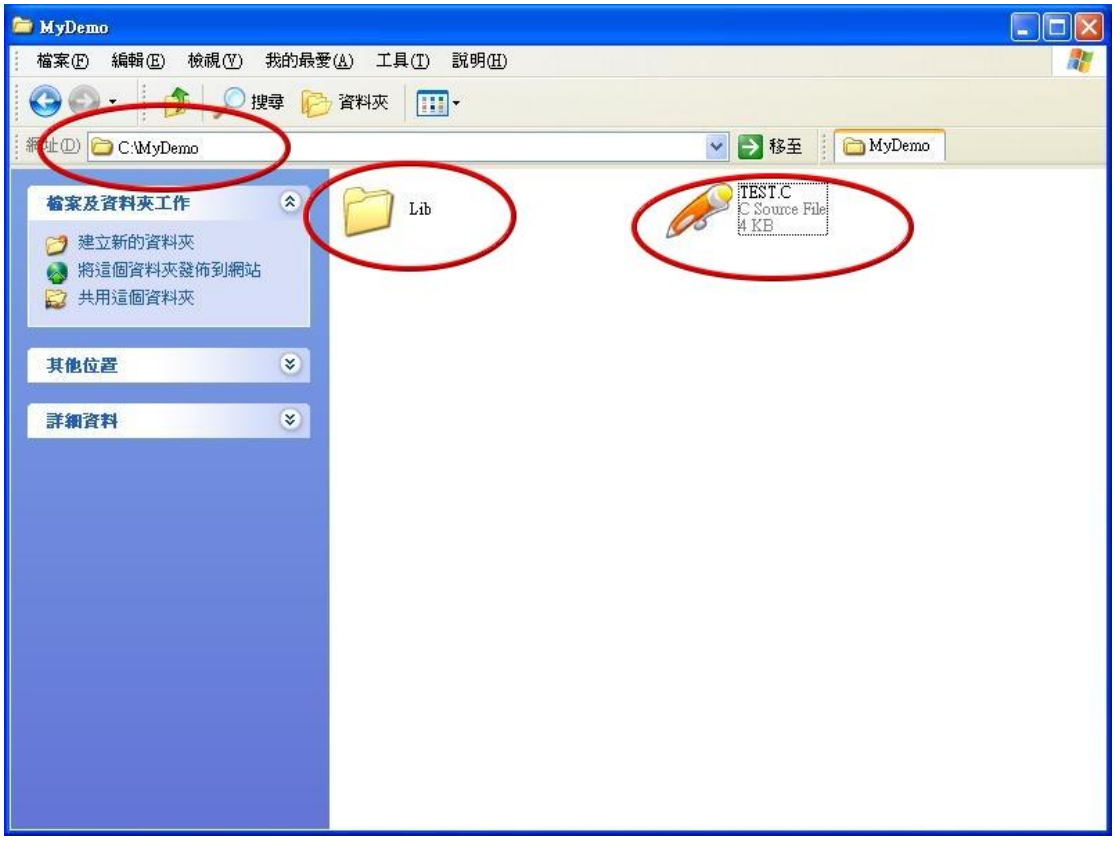
Return the current version number.

7. Program Download Procedure

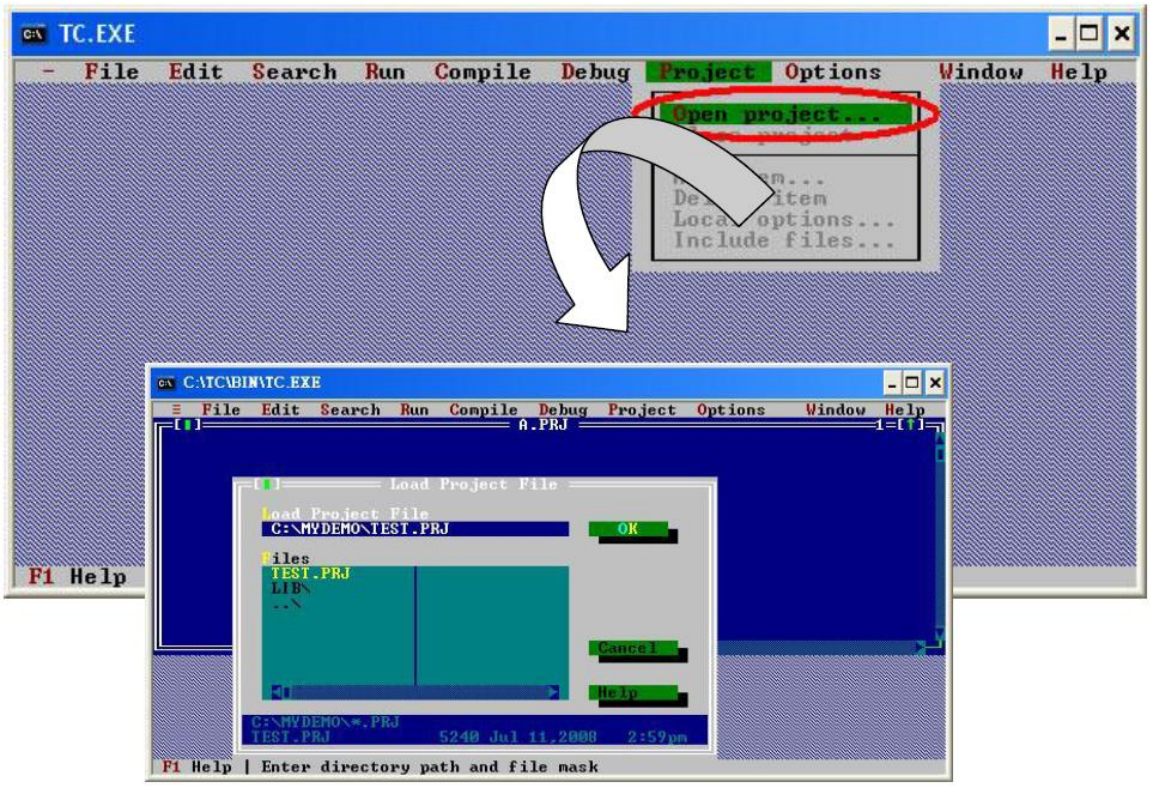
Here, it is considered that how to build an execution file and how to run this program on the G-4513 series.

Library	Description
G4500.LIB	G-4513 and DI/O、AI functions
GSM_U2.LIB	GPRS functions
MMC_FS4.LIB	MMC/SD functions
TCP_DM32.LIB	Ethernet functions
LCD.LIB	LCD functions

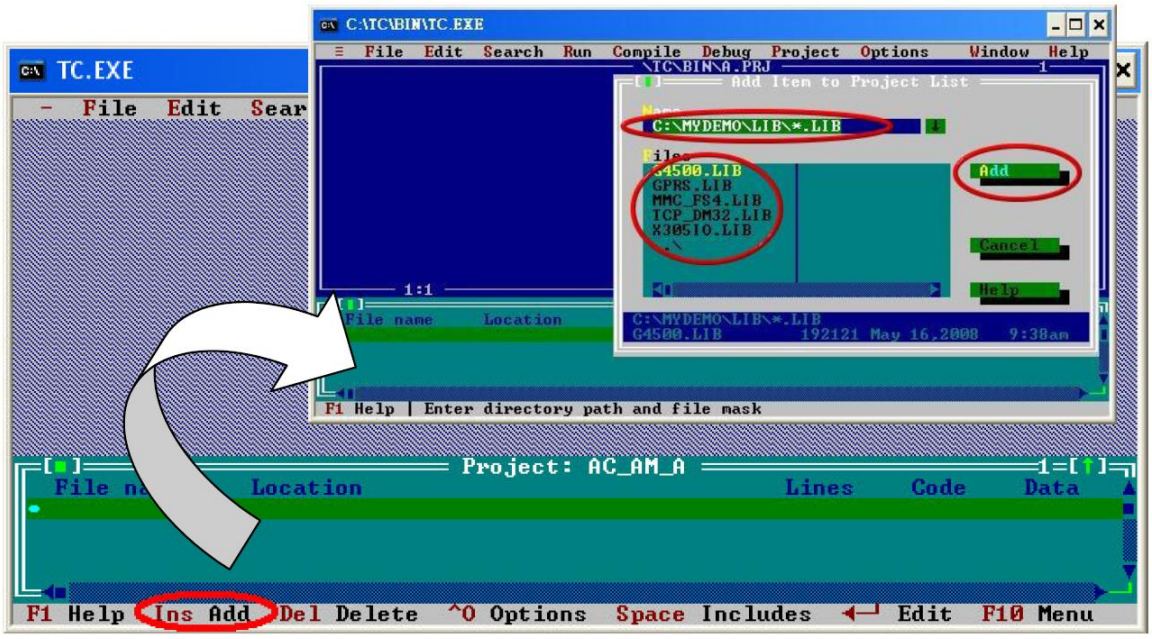
Step 1：Create a folder name "MyDemo" in the C disk, and copy the lib folder and users program into the MyDemo folder.



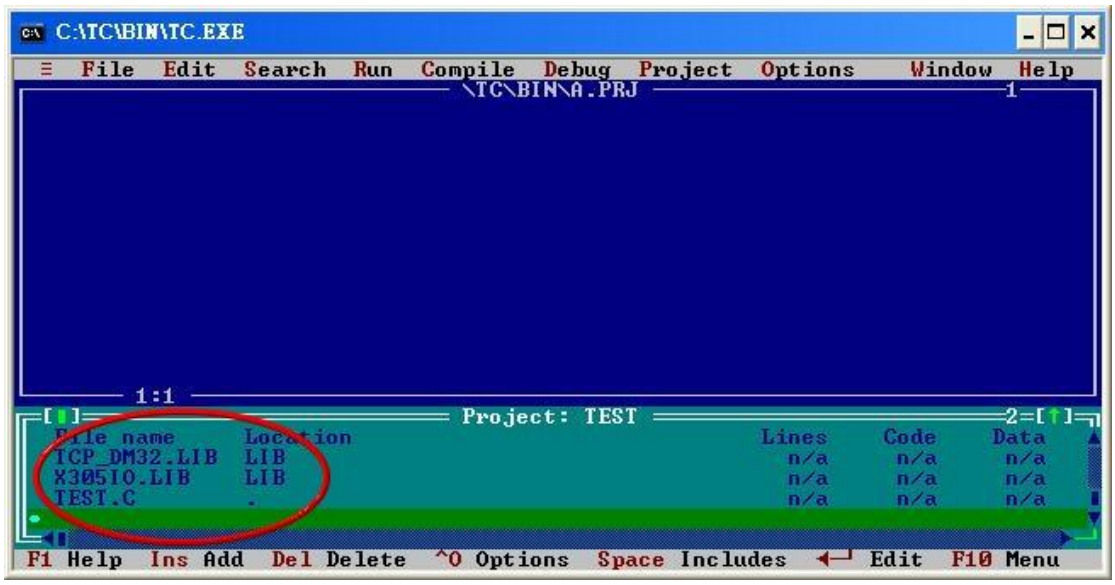
Step 2：Run the TC++1.01development. Click the "Project\Open project…" create new project named "TEST.PRJ".



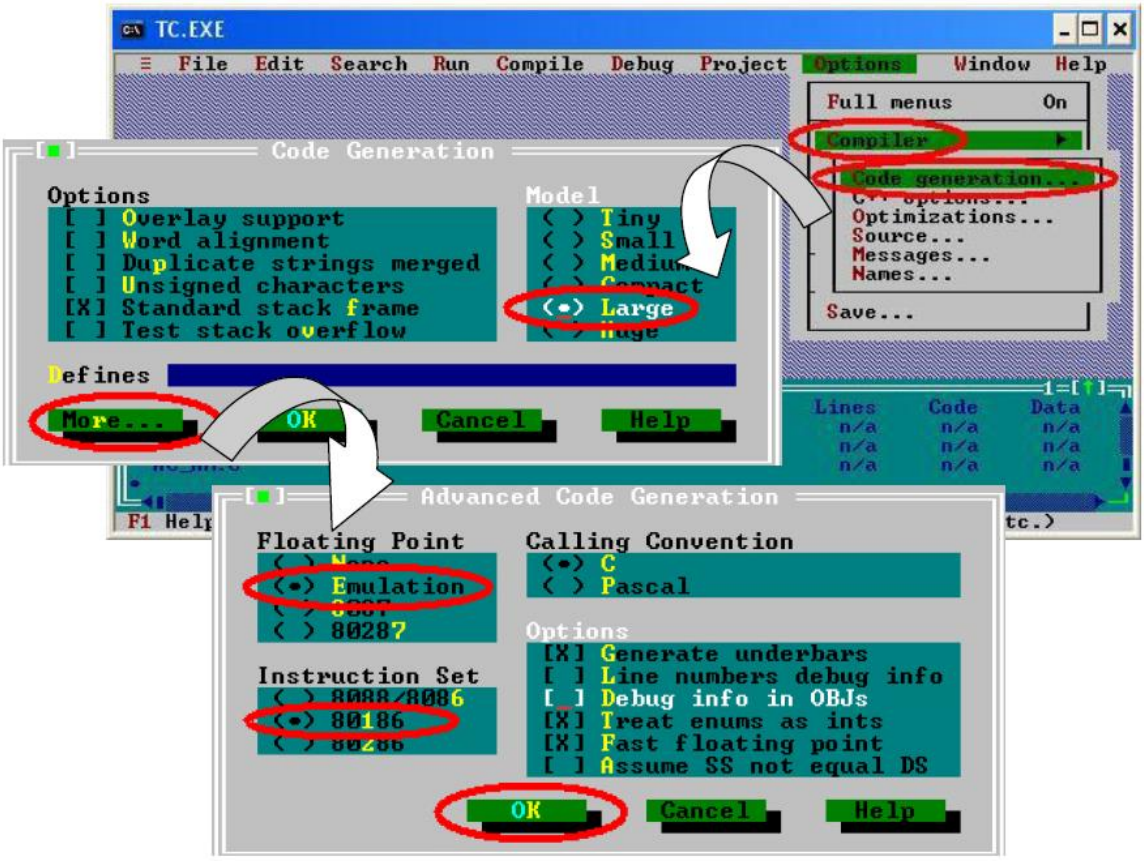
Step 3：Use the "Add" function to add the library file into MyDemo project.



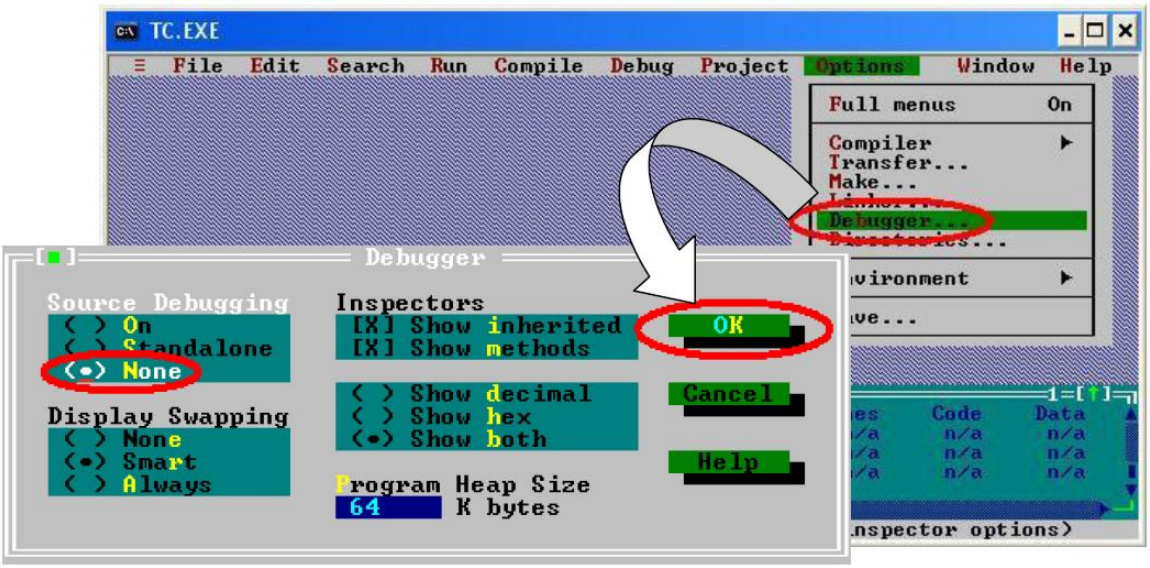
Step 4：Following the step3, add another library and TEST.C into MyDemo project.


C:\TC\BIN\TC.EXE
File Edit Search Run Compile Debug Project Options Window Help
\TC\BIN\A.PRJ
1:1
Project: TEST
File name Location Lines Code Data
TCP_DM32.LIB LIB n/a n/a n/a
X305IO.LIB LIB n/a n/a n/a
TEST.C . n/a n/a n/a
F1 Help Ins Add Del Delete ^O Options Space Includes Edit F10 Menu

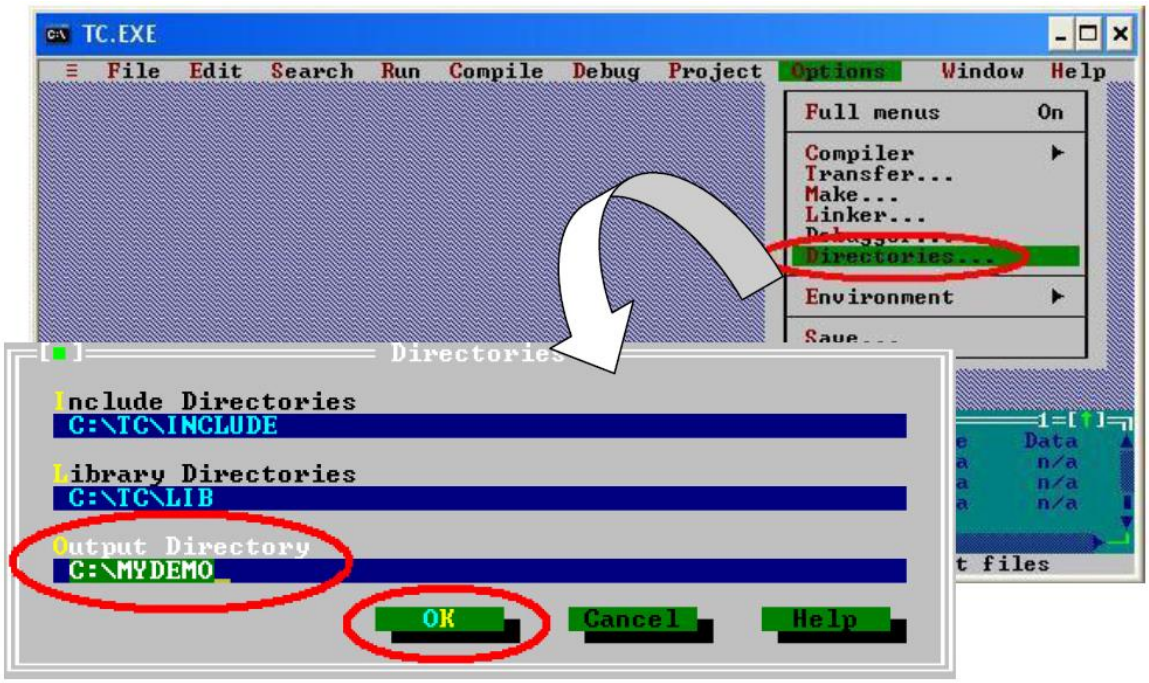
Step 5：Click the "Options/Compiler/Code generation…" to set the compile mode to the large mode. Click "More…" to set the "Floating point" and "Instruction Set" parameters. The Emulation and 80186 will be used respectively. Then, click OK button to save the configuration.



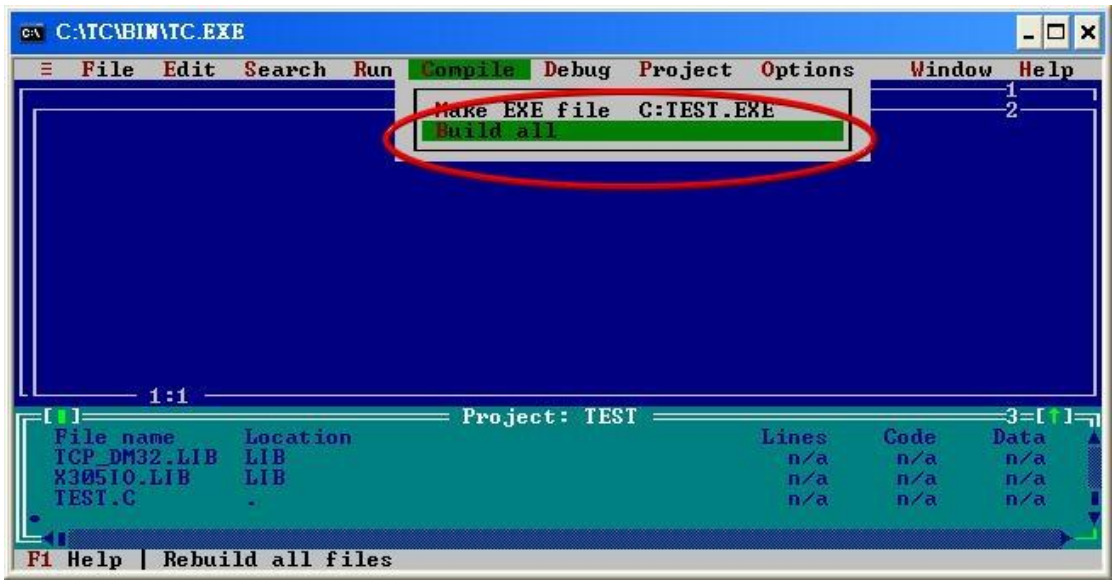
Step 6：Click the "Option/Debugger... " to set the "Source Debugging" parameter. Here, select the "None" for the "Source Debugging".



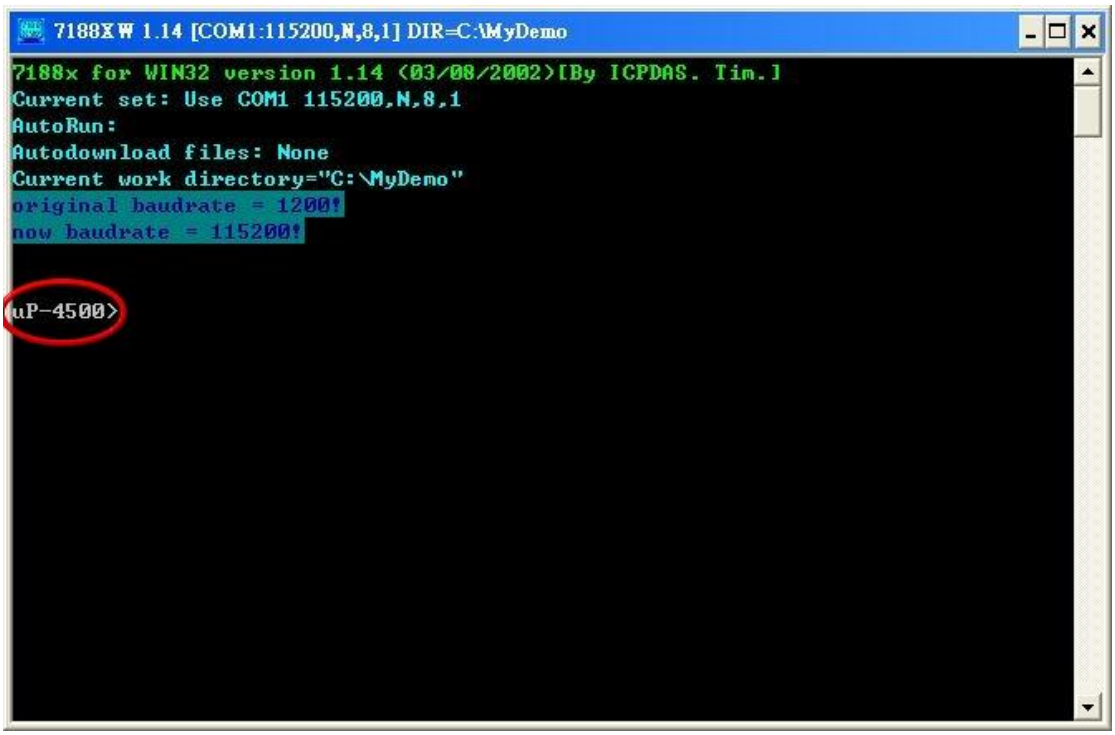
Step 7：Click the "Option/Directories... " to set the "Output Directory" parameter. Here, set the "C:\MyDemo" for the "Output Directory" parameter.



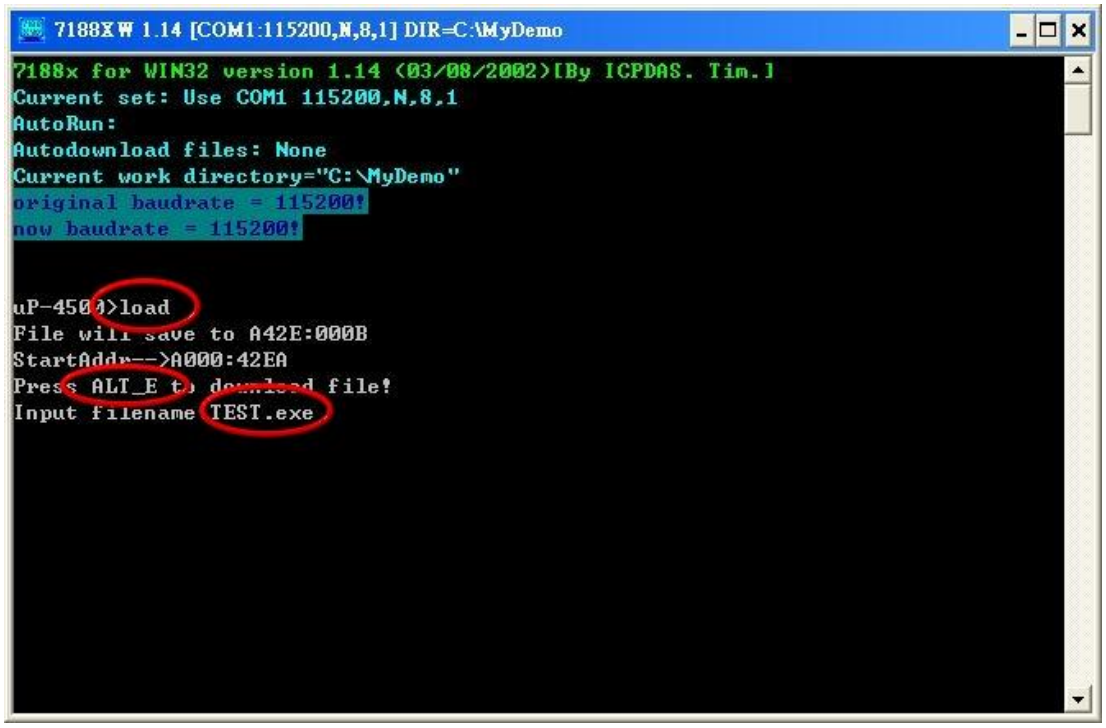
Step 8：After finishing all the parameters setting, click the "Compile/build all" to produce the execution file name "TEST.exe".


C:\TC\BIN\TC.EXE
File Edit Search Run Compile Debug Project Options Window Help
Make EXE file C:TEST.EXE
Build all
1:1
Project: TEST
File name Location Lines Code Data
TCP_DM32.LIB LIB n/a n/a n/a
X305IO.LIB LIB n/a n/a n/a
TEST.C . n/a n/a n/a
F1 Help | Rebuild all files

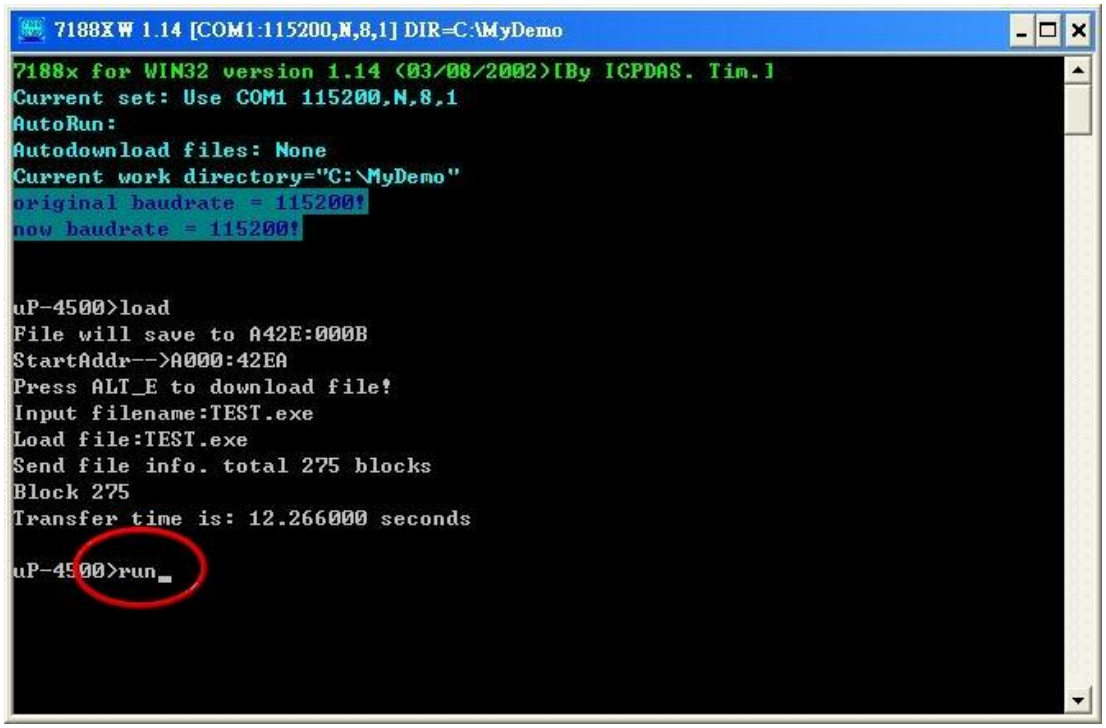
Step 9：Copy the file 7188XW.exe into the MyDemo folder. Then, double-click the 7188XW.exe file.The 7188XW.exe can be found in the Osimage folder. And G-4513 series COM1 connected to the PC RS-232.



Step 10：Key the command, "load" in the 7188xw.exe program. Then, follow the hint command to press "Alt+E" and input the file name, "TEST.exe", to download the execution file.



Step 11：After finishing the download procedure, key in the command, "run", to implement the execution file, "TEST.exe".



8. Revision History

Revision	Date	Author	Description
1.0	2014/09/01	William	Release version